

1 CLAIMS

2

3 1. An electronic document editor, comprising:
4 a default event handler to process editing events;
5 a designer extensibility mechanism to communicate with an extension
6 coupled with the editor, the extension being configured to process at least one of
7 the editing events; and
8 wherein the designer extensibility mechanism provides the editing events to
9 the extension prior to the default event handler processing the editing events.

10

11 2. The electronic document editor as recited in claim 1, further
12 comprising an extension interface having a pre-handle event method through
13 which the designer extensibility mechanism provides the events to the extension.

14

15 3. The electronic document editor as recited in claim 1, wherein the
16 designer extensibility mechanism further provides the editing events to the
17 extension after the default event handler processes the editing events.

1 4. The electronic document editor as recited in claim 3, further
2 comprising an extension interface having a pre-handle event method and a post-
3 handle event method, the designer extensibility mechanism providing the editing
4 events to the extension through the pre-handle event method prior to the default
5 event handler processing the editing events, and providing the editing events to the
6 extension through the post-handle event method after the default event handler has
7 processed the editing events.

8
9 5. The electronic document editor as recited in claim 1, wherein:
10 the extension is a first extension; and
11 the designer extensibility mechanism further provides the editing events to
12 a second extension after the editing events have been provided to the first
13 extension, and prior to the default event handler processing the editing events.

14
15 6. The electronic document editor as recited in claim 5, wherein the
16 designer extensibility mechanism further provides the editing events to the first
17 extension and the second extension after the default event handler processes the
18 editing events.

19
20 7. The electronic document editor as recited in claim 1, wherein:
21 the extension is a first extension;
22 the designer extensibility mechanism further provides the editing events to
23 a second extension; and

24
25

the designer extensibility mechanism further provides notice to the first extension of any action taken on an event by the second extension or the default event handler.

8. In an extensible editor having an editor extension coupled therewith, the editor having an edit designer interface comprising a pre-handle event method, the pre-handle event method comprising:

routing an editing event to the editor extension before the editor acts on the editing event; and

receiving notification from the editor extension indicating whether the editor should continue to process the editing event after the editing event has been routed to the editor extension.

9. The method as recited in claim 8, wherein the routing an editing event to the editor extension further comprises routing an event identifier to the editor extension, the event identifier uniquely identifying an editing event.

10. The method as recited in claim 8, wherein the routing an editing event to the editor extension further comprises routing an event object interface to the editor extension, the event object interface providing the editor extension with means to process the editing event.

1 **11.** In an extensible editor having an editor extension coupled therewith,
2 the editor having an edit designer interface comprising a post-handle event
3 method, the post-handle event method comprising:

4 routing an editing event to the editor extension after the editor acts on the
5 editing event; and

6 receiving notification from the editor extension indicating whether the
7 editor should continue to process the editing event after the editing event has been
8 routed to the editor extension.

9
10 **12.** The method as recited in claim 11, wherein the routing an editing
11 event to the editor extension further comprises routing an event identifier to the
12 editor extension, the event identifier uniquely identifying an editing event.

13
14 **13.** The method as recited in claim 11, wherein the routing an editing
15 event to the editor extension further comprises routing an event object interface to
16 the editor extension, the event object interface providing the editor extension with
17 means to process the editing event.

1 **14.** A system, comprising:
2 an extensible editor that processes editing events, the extensible editor
3 having an event routing controller and a default event handler;
4 an extension coupled with the extensible editor for processing the editing
5 events; and
6 wherein the event routing controller provides an editing event received by
7 the editor to the extension prior to providing the editing event to be processed by
8 the default event handler.

9
10 **15.** The system as recited in claim 14, wherein:
11 the extension is a first extension;
12 the system further comprises a second extension for processing the editing
13 events; and
14 the event routing controller provides the editing event to the second
15 extension prior to providing the event to the default event handler.

16
17 **16.** The system as recited in claim 14, wherein the event routing
18 controller provides the editing event to the extension after the default event
19 handler has processed the editing event.

1 **17.** The system as recited in claim 14, wherein:
2 the extension is a first extension;
3 the system further comprises a second extension for processing the editing
4 events;
5 the event routing controller routes editing events to the first extension, the
6 second extension and the default event handler; and
7 each event is notified of any action taken in response to the editing event by
8 the other extension or by the default event handler.

9
10 **18.** The system as recited in claim 14, wherein the event routing
11 controller further provides an event identifier uniquely identifying the editing
12 event.

13
14 **19.** The system as recited in claim 14, wherein the event routing
15 controller provides an event object interface to the extension to allow the
16 extension to access information regarding the editing event.

17
18 **20.** The system as recited in claim 14, wherein the editor further
19 comprises an edit designer interface that includes a pre-handle event method for
20 providing the event to the extension prior to the default event handler receiving the
21 event.

1 **21.** The system as recited in claim 14, wherein:

2 the editor further comprises an edit designer interface that includes a post-
3 handle event method for providing the editing event to the extension after the
4 default event handler has processed the event; and

5 the event routing controller is further configured to provide the editing
6 event to the extension through the edit designer interface.

7
8 **22.** The system as recited in claim 14, wherein:

9 the extension is a first extension;
10 the editor further comprises an edit designer interface that includes a post-
11 editor event notify method that is called by the event routing controller after the
12 editing event has been processed by the first extension, a second extension and the
13 default event handler to provide data to the first extension and the second
14 extension regarding actions taken in response the editing event.

15
16 **23.** A designer attached to an editor, comprising a pre-event handler that
17 processes an editing event from the editor before the editor processes the event.

18
19 **24.** The designer as recited in claim 23, further comprising a post-event
20 handler that processes the editing event after the editor processes the editing event.

21
22 **25.** The designer as recited in claim 23, wherein the pre-event handler
23 processes the editing event and notifies the editor to prevent further processing on
24 the event.

1 **26.** The designer as recited in claim 23, wherein the default event
2 handler responds to the editing event by notifying the editor to continue processing
3 the event.

4
5 **27.** The designer as recited in claim 23, further comprising a post-event
6 handler that processes the editing event after the editor processes the editing event,
7 and notifies the editor to prevent further processing on the event.

8
9 **28.** The designer as recited in claim 23, further comprising a post-event
10 handler that processes the editing event after the editor processes the event, and
11 notifies the editor to continue processing the editing event.

12
13 **29.** An editor that communicates with a first designer and a second
14 designer, comprising:

15 a default event handler; and

16 an edit designer interface that includes a pre-handle event method to
17 process an event before the default event handler processes the event, and a post-
18 handle event method to process the event after the default event handler has
19 processed the event.

20
21 **30.** The editor as recited in claim 29, further comprising a post-editor
22 event notify method that is called to notify the first extension of an action taken by
23 the second extension when processing the event.

1 **31.** The designer as recited in claim 23, further comprising a translate
2 accelerator method that translates commands received by the editor.

3
4 **32.** The designer as recited in claim 23, wherein the pre-handle event
5 method and the post-handle event method include an event ID parameter that
6 uniquely identifies the event.

7
8 **33.** The designer as recited in claim 23, wherein the pre-handle event
9 method and the post-handle event method include an event object interface that
10 allows the extensions to obtain information about the event.

11
12 **34.** An edit designer interface in an extensible editor, comprising:
13 a pre-handle event method used to send an editing event to a designer
14 attached to the editor prior to the editor processing the editing event; and

15 a post-handle event method used to send the editing event to the designer
16 after the editor has processed the editing event.

17
18 **35.** The edit designer interface as recited in claim 34, wherein:
19 the designer is a first designer;
20 the edit designer interface further comprises a post-editor event notify
21 method used to notify a second designer attached to the editor of an action taken
22 by the first designer when the first designer processed the editing event.

1 **36.** The edit designer interface as recited in claim 34, wherein the pre-
2 handle event method and the post-handle event method include an event ID
3 parameter that uniquely identifies the editing event.

4

5 **37.** The edit designer interface as recited in claim 34, wherein the pre-
6 handle event method and the post-handle event method include an event object
7 interface associated with the editing event through which the designer can obtain
8 information regarding the editing event.

9

10 **38.** A method for processing events in an extensible editor that
11 communicates with a first extension and a second extension, the method
12 comprising:

13 sending an event to the first extension; and
14 receiving a signal from the first extension indicating whether to continue
15 processing the event.

16

17 **39.** The method as recited in claim 38, further comprising notifying the
18 second extension about actions taken by the first extension in response to
19 receiving the event if the signal indicates that the event should not be processed
20 further.

21

22 **40.** The method as recited in claim 38, further comprising processing
23 the event if the signal indicates that processing should continue.

1 **41.** The method as recited in claim 38, further comprising sending the
2 event to the second extension if the signal indicates that processing should
3 continue.

4
5 **42.** The method as recited in claim 38, further comprising:
6 determining if the event is a command; and
7 if the event is a command, translating the command and withholding the
8 event from the extensions.

9
10 **43.** The method as recited in claim 38, further comprising:
11 determining if the event is a command; and
12 if the event is a command, translating the command and notifying the
13 extensions that the command has been processed.

14
15 **44.** A computer-readable medium having computer-executable
16 instructions that, when executed on a computer, perform the following steps:
17 detect an editing event;
18 routing the editing event to a designer;
19 receiving a response from the designer that indicates whether the editing
20 event was consumed by the first designer.

21
22 **45.** The computer-readable medium as recited in claim 44, further
23 comprising computer-executable instructions to perform the following step:
24 processing the editing event if the response from the designer indicates that
25 the editing event was not consumed by the first designer.

1
2 **46.** The computer-readable medium as recited in claim 44, wherein the
3 designer is a first designer, and further comprising computer-executable
4 instructions to perform the following step:

5 notifying a second designer that the first designer consumed the event if the
6 response from the first designer indicates that the first designer consumed the
7 event.

8
9 **47.** The computer-readable medium as recited in claim 44, wherein the
10 designer is a first designer, and further comprising computer-executable
11 instructions to perform the following step:

12 routing the event to the second designer if the response from the first
13 designer indicates that the first designer did not consume the event.